Discriminative Visual

Unifying Discriminative Visual Codebook Generation with Classifier Training for Object Category Recognition

Liu Yang  Michigan State University
Rong Jin    Michigan State University
Rahul Sukthankar Intel Research & Carnegie Mellon
Frederic Jurie    INRIA
Object Category Recognition

- sheep? ✗
- bus? ✓
- cat? ✗
- bicycle? ✓
- car? ✓
- cow? ✗
- dog? ✗
- horse? ✗
- mbike? ✓
- person? ✓
Standard Approach (adopted from text IR)

[Fei-Fei et al., 2005; Sivic et al., 2005; and many others]

Feature extraction and representation (e.g., SIFT)

Quantization + histogram

“Bag of visual words”

- sheep? ✗
- bus? ✓
- cat? ✗
- bicycle? ✓
- car? ✓
- cow? ✗
- dog? ✗
- horse? ✗
- mbike? ✓
- person? ✓

Classification (e.g., SVMs)
Codebook Construction by Clustering

Limitations:

• Universal dictionary → category independent
• Unsupervised clustering → ignores labeling information
• Every SIFT feature forced into one cluster → failure to capture partial similarity
• Difficulty in deciding the number of clusters → wrong choice leads to poor dictionaries
Limitation (II):

Every SIFT feature forced into one cluster → failure to capture partial similarity

Difficulty in deciding the number of clusters → wrong choice leads to poor dictionaries
Codebook Construction by Clustering

Limitations (cont.):
Codebook may not be discriminative to differentiate object categories
Understanding Clustering

- Clustering is a special coding
Understanding Clustering

- Clustering is a special coding
  - One and only one bit is on
- More general coding
  - Error Correcting Output Code (ECOC)
Understanding Clustering

• Clustering is a special coding
  – One and only one bit is on
• Our approach: coding by thresholded projections
Understanding Clustering

• Clustering is a special coding
  – One and only one bit is on
• Our approach: coding by thresholded projections

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x5</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x6</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Understanding Clustering

• Clustering is a special coding
  – One and only one bit is on
• Our approach: coding by thresholded projections

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x₁</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x₂</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x₃</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x₄</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x₅</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x₆</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Understanding Clustering

• Clustering is a special coding
  – One and only one bit is on
• Our approach: coding by thresholded projections
Understanding Clustering

• Clustering is a special coding
  – One and only one bit is on

• Our approach: coding by thresholded projections
  – Non-orthogonal codes – chosen for maximal class separation
  – Key questions: how to select the projections P and thresholds b?

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>x3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>x6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Anatomy of a Visual Bit

\[ g_k(x, y) = I(x^T w_k^y - b_k^y) = \begin{cases} 
1 & x^T w_k^y > b_k^y \\
0 & x^T w_k^y \leq b_k^y 
\end{cases} \]

(learned)

“Is this feature relevant to the `bus` category?”

- Weakly-supervised learning of visual bits
- Applying visual bits to object category recognition

Yang, Jin, Sukthankar, Jurie – CVPR 2008
## Image Classification using Visual Bits

### Category $a$

<table>
<thead>
<tr>
<th></th>
<th>$g_1(x,a)$</th>
<th>$g_2(x,a)$</th>
<th>...</th>
<th>$g_l(x,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>1</td>
<td>..</td>
<td>...</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>0</td>
<td>..</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_n$</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

feature-level representation
Image Classification using Visual Bits

<table>
<thead>
<tr>
<th>Category</th>
<th>$g_1(x,a)$</th>
<th>$g_2(x,a)$</th>
<th>...</th>
<th>$g_l(x,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_n$</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

The table above represents the classification of images into different categories based on visual bits. The image representation is shown to the right of the table.
Image Classification using Visual Bits

<table>
<thead>
<tr>
<th>Category $a$</th>
<th>$g_1(X,a)$</th>
<th>$g_2(X,a)$</th>
<th>$\ldots$</th>
<th>$g_T(X,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

Classifier for Cat. $a$

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X,a)$$
Image Classification using Visual Bits

Category $a$

<table>
<thead>
<tr>
<th>$g_1(X,a)$</th>
<th>$g_2(X,a)$</th>
<th>...</th>
<th>$g_T(X,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>...</td>
</tr>
</tbody>
</table>

Classifier for Cat. $a$

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a)$$

Category $b$

<table>
<thead>
<tr>
<th>$g_1(X,b)$</th>
<th>$g_2(X,b)$</th>
<th>...</th>
<th>$g_T(X,b)$</th>
</tr>
</thead>
</table>

Classifier for Cat. $b$

$$f_b(X) = \sum_{k=1}^{T} \alpha_k g_k(X, b)$$

Category $z$

<table>
<thead>
<tr>
<th>$g_1(X,z)$</th>
<th>$g_2(X,z)$</th>
<th>...</th>
<th>$g_T(X,z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classifier for Cat. $z$

$$f_z(X) = \sum_{k=1}^{T} \alpha_k g_k(X, z)$$

Learn visual bit functions $g(x, a)$ and weights $\alpha$ together

Unify code generation with discriminative classifier

Yang, Jin, Sukthankar, Jurie – CVPR 2008
Image Classification using Visual Bits

<table>
<thead>
<tr>
<th>Category a</th>
<th>Classifier for Cat. a</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1(X,a)$</td>
<td>$f_{a_1}(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a)$</td>
</tr>
<tr>
<td>$g_2(X,a)$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>$g_T(X,a)$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category b</th>
<th>Classifier for Cat. b</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1(X,b)$</td>
<td></td>
</tr>
<tr>
<td>$g_2(X,b)$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>$g_T(X,b)$</td>
<td></td>
</tr>
</tbody>
</table>

Generalizes to nonlinear classifier (can be implemented using standard SVM)

$$f_{a}(X) = \sum_{i=1}^{N} \alpha_i k(\vec{g}(X, a), \vec{g}(X_i, a))$$

$k(x, x) : \mathbb{R}^T \times \mathbb{R}^T \rightarrow \mathbb{R}$: kernel function

$\vec{g}(X, a) = (g_1(X, a), \ldots, g_T(X, a))$: visual bit vector

$g_k(X, z)$
Standard Approach

Feature extraction and representation (e.g., SIFT)

Quantization + histogram

Classification (e.g., SVMs)

“Bag of visual words”

- sheep? ✗
- bus? ✓
- cat? ✗
- bicycle? ✓
- car? ✓
- cow? ✗
- dog? ✗
- horse? ✗
- mbike? ✓
- person? ✓

SVM SVM SVM SVM SVM SVM SVM SVM SVM SVM

FreeFoto.com

Yang, Jin, Sukthankar, Jurie – CVPR 2008
Unified Approach

Feature extraction and representation (e.g., SIFT)

class-specific visual bits

How to learn this discriminative representation in a weakly-supervised setting?

class-specific visual bits

• sheep? ✗
• bus? ✓
• cat? ✗
• bicycle? ✓
• car? ✓
• cow? ✗
• dog? ✗
• horse? ✗
• mbike? ✓
• person? ✓
Learning Visual Bits
Optimization Framework

- Given visual bit functions $g(x, a)$ and weights $\alpha$, how to measure if they are able to classify image $X=(x_1, \ldots, x_n)$ into cat. $(y_1, y_2, \ldots, y_K)$

**Challenge**
Which features correspond to which categories, or do not correspond to any category of interest at all?
Learning Visual Bits
Optimization Framework

- Given visual bit functions $g(x, a)$ and weights $\alpha$, how to measure if they are able to classify image $X=(x_1, \ldots, x_n)$ into cat. $(y_1, y_2, \ldots, y_k)$

\[
\begin{align*}
  f(x_3, y_1) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_1) \\
  f(x_3, y_2) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_2) \\
  f(x_3, y_3) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_3)
\end{align*}
\]
Learning Visual Bits
Optimization Framework

- Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X=(x_1, \ldots, x_n) \) into cat. \( (y_1, y_2, \ldots, y_k) \)

\[
\begin{align*}
    f(x_3, y_1) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_1) \\
    f(x_3, y_2) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_2) \\
    f(x_3, y_3) &= \sum_{k=1}^{T} \alpha_k g_k(x_3, y_3) \\
    e(x_3, y_i) &= \frac{\exp(f(x_3, y_i))}{\sum_{z=1}^{m} \exp(f(x_3, z))}
\end{align*}
\]
Relevant Visual Bits Localize Concepts
Learning Visual Bits
Optimization Framework

• Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X=(x_1, \ldots, x_n) \) into cat. \( (y_1, y_2, \ldots, y_K) \)

Loss function for image \( X \)

\[
l(X, y_1) = \frac{n}{\sum_{j=1}^{n} e(x_j, y_1)}
\]
Learning Visual Bits
Optimization Framework

• Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X=(x_1, \ldots, x_n) \) into cat. \( (y_1, y_2, \ldots, y_K) \)

\[
\text{Loss function for image } X \\
l(X, y_1) = \frac{n}{\sum_{j=1}^{n} e(x_j, y_1)}
\]

1. Diminishing rewards
2. Relation to exponential loss
Learning Visual Bits
Optimization Framework

• Given visual bit functions $g(x, a)$ and weights $\alpha$, how to measure if they are able to classify image $X=(x_1, \ldots, x_n)$ into cat. $(y_1, y_2, \ldots, y_K)$

Loss function for image $X$

$$l(X, y) = \sum_{y \in y} l(X, y)$$

$$l(X, y_1) = \frac{n}{\sum_{j=1}^{n} e(x_j, y_1)}$$

Loss function for the image collection

$$\mathcal{L}(x_{1:T}, g_{1:T}) = \sum_{i=1}^{N} l(X_i, y_i)$$
Learning Visual Bits
Optimization Framework

Given a collection of training images

\[ \mathcal{T} = \{(X_i, y_i), i = 1, \ldots, N\} \]

Find optimal visual bits and combination weights by solving

\[
\min_{g_{1:T}, \alpha_{1:T}} \mathcal{L}(\alpha_{1:T}, g_{1:T}) = \sum_{i=1}^{N} l(X_i, y_i)
\]

Overview of optimization algorithm (reminiscent of boosting)

- Iterative approach: learn one visual bit \((g)\) and weight \((\alpha)\) at a time
- Employ bound optimization to decouple \(g\) and \(\alpha\)

[details in paper and supplementary material]
Results on PASCAL 2006
(AUR with 100 training examples)

• Follows methodology from [Marszalek & Schmid, 2006]
• Baselines
  – Standard: K-means (k=1000) + SVM (\(\chi^2\) kernel)
  – Discriminative: Extremely Randomized Clustering Forests

<table>
<thead>
<tr>
<th>Class</th>
<th>KM-SVM</th>
<th>ERCF</th>
<th>Our Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>sheep</td>
<td>0.551 ± 0.046</td>
<td>0.747 ± 0.017</td>
<td>0.842 ± 0.008</td>
</tr>
<tr>
<td>bus</td>
<td>0.618 ± 0.030</td>
<td>0.708 ± 0.024</td>
<td>0.930 ± 0.005</td>
</tr>
<tr>
<td>cat</td>
<td>0.697 ± 0.011</td>
<td>0.753 ± 0.015</td>
<td>0.759 ± 0.016</td>
</tr>
<tr>
<td>bicycle</td>
<td>0.750 ± 0.026</td>
<td>0.744 ± 0.021</td>
<td>0.782 ± 0.021</td>
</tr>
<tr>
<td>car</td>
<td>0.654 ± 0.043</td>
<td>0.731 ± 0.019</td>
<td>0.875 ± 0.007</td>
</tr>
<tr>
<td>cow</td>
<td>0.519 ± 0.026</td>
<td>0.751 ± 0.026</td>
<td>0.790 ± 0.017</td>
</tr>
<tr>
<td>dog</td>
<td>0.670 ± 0.011</td>
<td>0.706 ± 0.026</td>
<td>0.761 ± 0.012</td>
</tr>
<tr>
<td>horse</td>
<td>0.503 ± 0.016</td>
<td>0.712 ± 0.025</td>
<td>0.671 ± 0.009</td>
</tr>
<tr>
<td>motor</td>
<td>0.496 ± 0.017</td>
<td>0.733 ± 0.019</td>
<td>0.782 ± 0.013</td>
</tr>
<tr>
<td>person</td>
<td>0.551 ± 0.035</td>
<td>0.729 ± 0.015</td>
<td>0.722 ± 0.007</td>
</tr>
</tbody>
</table>
Conclusion

• Unify codebook construction + classifier training
  – Generate codebooks by iterative projection
  – Efficiently learn projection and weights together

• Impact on object category recognition
  – Learns better representations with limited training data
  – No parameters to tune